Truck Platooning: week 2

Changyao Zhou, Jiaxin Pan

December 20, 2021

1 Introduction of the project

The goal of this project is to chase a chosen target vehicle by estimating the steering angle and throttle input of the ego-vehicle, given the velocity of both vehicles.

The control values (steering angle and throttle) computed by Model Predictive Control (MPC) are used as ground-truth. MPC is an advanced method of process control that is used to control a process while satisfying a set of constraints[WIK]. It takes a prediction horizon instead of a single time step into account, and aims to get an optimal control result minimizing the objective function within the prediction horizon. For learning the MPC controller, we take [MPC] as reference.

The project is designed to be realized in the following steps:

1). Control the ego-vehicle with Model Predictive Control (MPC) in the Carla simulator

2). Given object velocity, ego-velocity and RGB image, train a neural network to control the ego-vehicle by using the MPC output as ground-truth

3). Further improvement by changing the input of the neural network

2 Overview of this week

Last week, we first implement the MPC control in our own model, but there's still some problem when integrating it into Carla simulator. In this week, we continued to implement the MPC controller in the Carla simulator to let the ego-vehicle follow the trajectory of the target vehicle.

Since we stick to the 'offline' mode, we first tried to adjust the plant model to fit the Carla simulator, and then generate a series of control values, steering angle and throttle, with MPC controller. After that, we put the control values back into Carla simulator to implement car following. Also, we have added simultaneous visualization of both camera image and location plot.

All the videos are stored under https://syncandshare.lrz.de/getlink/fiXpyEuXj2Tw7LhZzG7eYMKf/week2.

3 Design the Cost Function

3.1 The Safe Distance

For the cost function, we added the dynamic safe distance S between two cars depend of the velocity of the target vehicle. L is the length of the ego-vehicle.

$$S = L * (1 + v_{target}/5) \tag{1}$$

The parameter 5 is chosen since the speed of the target vehicle reaches the balance point with 5 m/s, based on the data we collected, 80% of the time the target vehicle drives with this speed. With the equation (1), the safe distance should be twice the length of the vehicle when the target vehicle has a constant velocity 5 m/s.

3.2 The Yaw Angle

The yaw angle in Carla simulator can not be matched perfectly to our simple model. The output yaw angle from Carla simulator has a range of [-π, π], while in the simple model it's [0, 2π]. This may lead to a problem that a very sharp turn around 360° is detected in the simple model when the yaw of the target vehicle changes from -179.5° to 179.5°, which is actually only a slight turn of 1°.

To solve this problem, we converted the range of the output yaw angle from Carla simulator to the range [0, 2π] with the equation (2):

$$yaw_{ref} = (yaw_{ref} + 2 * pi)\%(pi * 2) \tag{2}$$

This lead to however another problem, for the original yaw angle -0.7°, it is converted to 359.3°, and for 0.1° it is converted to 0.1°. If the yaw of the target vehicle changed from -0.7° to 0.1°, a very sharp turn with around 360° is detected, while the actual turn is only 0.8°. If the current yaw of ego vehicle is -0.7°, an extremely large cost will be added, which is unreasonable. This is solved by modifying the cost function of the yaw angle:

$$yaw_{ego} = abs(yaw_{ego} - pi) \tag{3}$$

$$yaw_{ref} = abs(yaw_{ref} - pi) \tag{4}$$

$$cost+ = abs(yaw_{ego} - yaw_{ref}) * *2 \tag{5}$$

With equation (3), (4) and (5), no longer the difference between the yaw angle of ego vehicle and target vehicle is computed. Instead, first their distance to 180° is computed separately and then the difference between these two distances are computed. For 359.3°, its distance to 180° is 179.3° and for 0.1° the distance is 180.1°. Therefore, the cost is reduced to 0.8°, which is much smaller in comparison to near 360°.

4 Design the Plant Model

4.1 Vehicle Startup Phase

We find that during the first several frames of startup phase, the car with autopilot mode in Carla has throttle = 0.69 but it remains still. And after a few frames, it starts with a relative large acceleration. To simulate this process, we set the acceleration = 0 during the first frames, and give the car a relative large acceleration of startup.

4.2 Vehicle Acceleration Phase with Different Gear

To refine the vehicle acceleration process in Carla, we find there's an attribute describing the discrete relationship between the throttle and acceleration, gear. With gear = 0, the car has the acceleration = 0. With gear = 1, the car with the same throttle has a larger acceleration. With gear = 2, the car has a smaller acceleration than with gear = 1. And the switch of gear happens at specific time steps. To model this process, the plant model is shown as follows. The input control values are throttle and steer angle.

steering angle of the front wheels

$$beta = \textbf{steer angle} * 1.62 \tag{6}$$

startup phase with gear 0

$$a_t = 0 \tag{7}$$

$$v_{t+1} = v_t \tag{8}$$

acceleration phase with gear 1

$$a_t = \textbf{throttle} * 11.9 \tag{9}$$

$$v_{t+1} = v_t + a_t * dt - v_t/11.65 \tag{10}$$

acceleration phase with gear 2

$$a_t = \textbf{throttle} * 3.75 \tag{11}$$
$$v_{t+1} = v_t + a_t * dt - v_t/36.95 \tag{12}$$

The rest part remains the same as last week.

$$x_{t+1} = x_t + v_t * dt * cos(psi_t) \tag{13}$$
$$y_{t+1} = y_t + v_t * dt * sin(psi_t) \tag{14}$$
$$psi_{t+1} = psi_t + v_t * dt * tan(beta)/length \tag{15}$$

Where x_t and y_t represent the current x and y location of the ego-vehicle, v_t is the current velocity, psi_t is the current yaw angle and dt is the time interval between two frames.

To adjust the parameters in the plant model above, we first collect the data like location, velocity, control values of a single autopilot trajectory. The correctness of the plant model is tested by comparing the velocity collected from autopilot mode and the velocity computed by plant model. If our model could generally describe the process in Carla, ideally these two velocities would be close to each other.

5 Offline and Online Mode of MPC Controller

Here we used the offline mode, which means we should first collect the data such as trajectory, velocity of the target vehicle from Carla. Then we could use MPC controller to generate the control values for ego-vehicle, taking the trajectory of the target within the following 15 time steps into account. After that, we could put the control values into the Carla simulator and test if the ego-vehicle controlled by offline MPC could follow the target and keep a specific distance successfully.

The main reason for not using online MPC is that with only one-step current trajectory of target vehicle as input of MPC controller, the ego-vehicle couldn't chase the target vehicle, even if it doesn't reach its speed limit. Thus, what is the reason of not able to chase, and how to implement the online MPC in Carla are still problems.

6 Conclusion and future work

In this week, we first tried to apply MPC controller to Carla simulator in a relatively simple trajectory. As a result, the ego-vehicle can successfully follow the target vehicle with a specific distance with offline MPC. The outcome is still not perfect espeacially with sharp turns. The video is stored under link https://syncandshare.lrz.de/getlink/fiBFmrCt7x2nFQoh11NhVkeb/car_following_town04.mp4. However, there's a serious problem that adjusting plant model is super complex and time-consuming. And the plant model fitting this trajectory might be not compatible with other trajectories or other maps. Thus, it's difficult to collect further dataset for training models in this way. So We wonder, if there's any better way to fit MPC into Carla simulator more generally.

References

[MPC] Model Predictive Control. https://github.com/WuStangDan/mpc-course-assignments.

[WIK] Model Predictive Control. https://en.wikipedia.org/wiki/Model_predictive_control.